dwaps / **Etape-1.md**

Last active 2 months ago • Report abuse

Star

Code Revisions 7 Forks 1

Git : Procédure pour débutant

Etape-1.md

Etape 1 : initialiser un dépôt (projet) git

Créer un dossier pour le futur projet

Dans le terminal, depuis le dossier utilisateur (home) :

```
michael@dwaps-formation:~$ cd Desktop/mon-super-projet
michael@dwaps-formation:~$ git init
Initialized empty Git repository in C:/Users/conta/Desktop/tests/.git/
```

Vérifier le nouvel état du dépôt

Pour voir comment Git signale sa prise en charge des ajouts, on peut vérifier l'état du dépôt :

```
michael@dwaps-formation:~$ git status
On branch master

Initial commit

nothing to commit (create/copy files and use "git add" to track)
```

Etape-2.md

Etape 2 : Ajouter les premières modifications à l'index de Git

Créer un premier fichier

Ne pas oublier de le remplir un peu...

```
touch index.html && notepad index.html
```

Vérifier l'état courant du dépôt

Git s'est apperçu qu'il y a un nouveau fichier. Il faut bien comprendre que ce qui intéresse Git ce sont les *modifications* : autrement dit un nouveau fichier EST une modification et c'est pour cela qu'il réagit.

```
git status
On branch master

Initial commit

Untracked files:
  (use "git add <file>..." to include in what will be committed)

        index.html
```

Ajouter le fichier à l'index de Git

Dès que le fichier est ajouté à l'index de Git, ce dernier suivra ses modifications (ajout/édition/suppression de code). Pour Git, ce sont les lignes qui compte : si un seul mot est changé, il considèrera que toute la ligne a été modifiée.

```
git add index.html
```

Vérifier à nouveau l'état du dépôt

Cela permettra de constater que Git a effectivement ajouté votre fichier à son index et qu'il est prêt à sauvegarder toutes les modifications que vous effectuerez.

```
michael@dwaps-formation:~$ git status
On branch master

Initial commit

Changes to be committed:
  (use "git rm --cached <file>..." to unstage)

        new file:   index.html
```

Recommencez !

Vous pouvez refaire les étapes précédentes pour mieux apprécier le comportement de Git...

```
michael@dwaps-formation:~$ notepad index.html
michael@dwaps-formation:~$ touch style.css && notepad style.css
michael@dwaps-formation:~$ git add index.html style.css
michael@dwaps-formation:~$ git status
On branch master

Initial commit

Changes to be committed:
  (use "git rm --cached <file>..." to unstage)

        new file:   index.html

Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git checkout -- <file>..." to discard changes in working directory)

        modified:   index.html

Untracked files:
  (use "git add <file>..." to include in what will be committed)

        style.css

michael@dwaps-formation:~$ git add .
michael@dwaps-formation:~$ git status
On branch master

Initial commit

Changes to be committed:
  (use "git rm --cached <file>..." to unstage)

        new file:   index.html
        new file:   style.css
```

Etape-3.md

Etape 3 : Faire son premier commit

Un commit, en gros c'est quoi ?

Un *commit* est une sauvegarde de modification(s) : lorsque le code a été modifié et que tout marche comme sur des roulettes, il faut faire une nouvelle sauvegarde pour **geler** le projet.

Après modification du code du projet, c'est l'heure de sauvegarder !

La commande `git commit` est accompagné d'un tag permettant de **coller** un bref message au commit.

```
michael@dwaps-formation:~$ git commit -m "Mon premier commit"
[master (root-commit) 0b1b4af] Mon premier commit
 2 files changed, 16 insertions(+)
 create mode 100644 index.html
 create mode 100644 style.css
```

Notion importante !

Un commit doit TOUJOURS être précédé d'un ajout de la modification à l'index de Git :

```
michael@dwaps-formation:~$ mkdir mon-dossier && touch fichier-1.sh fichier-2.png
michael@dwaps-formation:~$ git add fichier-1.sh fichier-2.png mon-dossier/
michael@dwaps-formation:~$ git commit -m "Modification de 3 fichiers"
[master e5a2ede] Modification de 3 fichiers
 3 files changed, 0 insertions(+), 0 deletions(-)
 create mode 100644 fichier-1.sh
 create mode 100644 fichier-2.png
 create mode 100644 mon-dossier/fichier-3.mp3

 michael@dwaps-formation:~$ git log
 commit e5a2ede (HEAD -> master)
Author: Michael Cornillon <contact@dwaps.fr>
Date:   Fri Dec 21 10:11:38 2018 +0100

    Modification de 3 fichiers

commit 0b1b4af
Author: Michael Cornillon <contact@dwaps.fr>
Date:   Fri Dec 21 10:08:31 2018 +0100

    Mon premier commit
```

(Notez que vous pouvez à tout moment voir l'historique de vos commits grâce à la commande `git log`).

Pour ajouter plus rapidement l'ensemble des modifications, on peut utiliser le . (répertoire courant) : `git add .` .

```
michael@dwaps-formation:~$ git add .
michael@dwaps-formation:~$ git commit -m "Modification de 3 fichiers"
```

Pour aller encore plus vite, on peut ajouter le tag `-a` à la commande `git commit` :

```
michael@dwaps-formation:~$ git commit -a -m "Modification de 3 fichiers"
```

Enfin, comme c'est possible dans la majorité des cas en bash, on peut **fusionner** les tags en un seul :

```
michael@dwaps-formation:~$ git commit -am "Modification de 3 fichiers"
```

Etape-4.md

Etape 4 : lié son dépôt local à un dépôt distant

Création d'un compte sur un serveur compatible

Avant tout, il faut ouvrir un compte sur un serveur qui comprend Git. Par exemple github, bitbucket ou gitlab.

Créer un dépôt sur le serveur

Ensuite, il faut créer un dépôt sur le serveur. Pour cela il suffit de suivre les indications qui sont assez intuitives. En général, on dispose d'un bouton +, on choisit d'ajouter un nouveau **repository** et il suffit seulement de lui donner un titre et de valider sans toucher à rien de plus.

Configuration du dépôt local

De retour sur votre ordinateur, dans votre projet, depuis le terminal...

Utiliser la commande `git remote` pour voir la liste des URLs vers lesquelles Git pourra envoyer (**pousser**) votre projet.

```
michael@dwaps-formation:~$ git remote -v
```

(L'option `-v` (mode verbeux) permet d'afficher plus de détail).

Evidemment, à ce stade aucun remote n'a été configuré. Pour le faire il suffit d'utiliser la même commande en ajoutant `add`, l'url et l'alias que vous choisirez. L'alias sert à éviter d'avoir à taper cette url à chaque fois que vous en aurez besoin. Pratique !

Remarque : l'url est de la forme `https://serveur.extension/mon-pseudo/mon-projet`.

```
michael@dwaps-formation:~$ git remote add github https://github.com/dwaps/cours-g
```

Il ne reste plus qu'à relancer la commande `git remote` pour voir apparaître la prise en compte par Git de l'url du remote (serveur distant destiné à héberger votre projet).

```
michael@dwaps-formation:~$ git remote
github

michael@dwaps-formation:~$ git remote -v
github  https://github.com/mon-pseudo/cours-git (fetch)
github  https://github.com/mon-pseudo/cours-git (push)
```

Etape-5.md

Etape 5 : pousser le dépôt local vers le dépôt distant

Apperçu du système de branche de Git

Pour le moment, il suffit de comprendre que Git peut gérer plusieurs branches. Une branche est simplement un enchaînement des commits que vous aurez effectués.

Rappel : En faisant un `git log`, vous pouvez voir à tout moment combien de commits vous avez fait. Tous ces commits font par défaut partie de la branche *master*.

```
michael@dwaps-formation:~$ git log
commit e5a2ede (HEAD -> master)
Author: Michael Cornillon <contact@dwaps.fr>
Date:   Fri Dec 21 10:11:38 2018 +0100

    Modification de 3 fichiers
```

```
commit 0b1b4af
Author: Michael Cornillon <contact@dwaps.fr>
Date:   Fri Dec 21 10:08:31 2018 +0100

    Mon premier commit
```

Envoie du code sur le serveur

Il suffit d'utiliser la commande `git push` en lui précisant le serveur sur lequel envoyer le code et la branche à envoyer. Pour le serveur, utilisez l'alias que vous avez choisi avec la commande `git remote add` !

```
michael@dwaps-formation:~$ git push github master
$  [master]  git push github master
Counting objects: 8, done.
Delta compression using up to 8 threads.
Compressing objects: 100% (5/5), done.
Writing objects: 100% (8/8), 842 bytes | 0 bytes/s, done.
Total 8 (delta 0), reused 0 (delta 0)
To github.com:dwaps/cours-git
 * [new branch]      master -> master
```

Remarque : vous devrez indiquez à chaque fois votre pseudo et votre mot de passe pour sécuriser l'action `push` . Notez que cela ne sécurise pas le transfert, pour cela il faudra une connexion en ssh (qui n'est pas traité ici).